MATHEMATISCHES INSTITUT
DER UNIVERSITÄT ZU KÖLN

Prof. Dr. A. Klawonn
J. Knepper, M. Sc.
J. Weber, M. Sc.

Wintersemester 2018/2019

21. November 2018

7. Übung zu Wissenschaftliches Rechnen I

Aufgabe 1: (7 Punkte)
Sei $a(\cdot,\cdot)$ die symmetrisch und positiv definite Bilinearform zur stationären Diffusionsgleichung mit inhomogener und positiver Koeffizientenfunktion auf dem Finite-Elemente-Raum $V^h = V^h(\Omega)$. Sei Γ das Interface zu einer Zerlegung von Ω. Mit $\mathcal{H}(u_\Gamma)$ bezeichnen wir die diskret harmonische Fortsetzung einer Funktion von Γ nach Ω:

$$u := \mathcal{H}(u_\Gamma) := \begin{pmatrix} -K_{II}^{-1} K_{I\Gamma} u_\Gamma \\ u_\Gamma \end{pmatrix}$$

Zeigen Sie, dass

$$a(u,u) = \min_{\substack{v \in V^h \\ v_\Gamma = u_\Gamma}} a(v,v)$$

erfüllt ist.

Aufgabe 2: $(2 + 3 + 1 + 3 = 9$ Punkte)

- Geben Sie die partielle Differentialgleichung an, sodass die Lösung der Diskretisierung gerade der diskret harmonischen Fortsetzung entspricht.
- Sei eine Partition der Eins auf dem Interface, auf Kanten und Eckknoten, durch $\theta_\mathcal{E}$ ($\mathcal{E} \subset \Gamma$) respektive $\theta_\mathcal{V}$ ($\mathcal{V} \subset \Gamma$) gegeben. Visualisieren Sie mit Matlab auf dem Gebiet $(0,1)^2$, zerlegt in 3×3 quadratische Teilgebiete, für das innere Gebiet („floating“) für einen Eckknoten und eine Kante Ihrer Wahl die diskret harmonische Fortsetzung von $\theta_\mathcal{E}$ bzw. $\theta_\mathcal{V}$.
- Wir bezeichnen mit $\theta_\mathcal{V}$ und $\theta_\mathcal{E}$ auch die Nullfortsetzung im Finite-Elemente-Raum $V^h(\Omega)$ von Γ nach Ω der entsprechenden Interfacewerte der Partition der Eins. Sei $v_h \in V^h$, gilt dann $\theta_\mathcal{V} v_h \in V^h$ bzw. $\theta_\mathcal{E} v_h \in V^h$?
- Sei Ω_5 das innere Gebiet („floating“) der 3×3-Gebietszerlegung. Geben Sie die Energie $a_{\Omega_5}(u,u) := \int_{\Omega_5} \nabla u \cdot \nabla u \, \mathrm{d}x$ für die Funktion

  $$u := \sum_{\mathcal{V} \subset \Gamma} \mathcal{H}(\theta_\mathcal{V}) + \sum_{\mathcal{E} \subset \Gamma} \mathcal{H}(\theta_\mathcal{E})$$

  an und beweisen Sie diese Gleichheit.

  Hinweis: Energieminimalität (Aufgabe 1)

Programmieraufgabe (FETI-DP): $(25 + 5 = 30$ Punkte) **Abgabe bis: 5. Dez.**
Im Folgenden betrachten wir das Modellproblem $-\Delta u = 1$ in $\Omega = (0,1)^2$ für $u|_{\partial\Omega} = 0$, diskretisiert mit einer strukturierten Dreieckszerlegung ($P1$). Sei Ω in $N \times N$, $N = 5$, quadratische Teilgebiete mit jeweils $2 \cdot 10^2$ Elementen zelegt.

a) Lösen Sie das FETI-DP-System, $F\lambda = d$, mit Ihrer PCG-Implementierung.

$$
\begin{aligned}
F &= B_B K_{BB}^{-1} B_B^T + B_B K_{BB}^{-1} \tilde{K}_{B\Pi} \tilde{S}_{\Pi\Pi}^{-1} \tilde{K}_{\Pi B} K_{BB}^{-1} B_B^T \\
d &= B_B K_{BB}^{-1} f_B - B_B K_{BB}^{-1} \tilde{K}_{B\Pi} \tilde{S}_{\Pi\Pi}^{-1} \left(\tilde{f}_\Pi - \tilde{K}_{\Pi B} K_{BB}^{-1} f_B \right) \\
\tilde{S}_{\Pi\Pi} &= \tilde{K}_{\Pi\Pi} - \tilde{K}_{\Pi B} K_{BB}^{-1} \tilde{K}_{B\Pi}
\end{aligned}
$$

Wählen Sie alle Eckknoten primal. Stellen Sie F nicht explizit auf, sondern schreiben Sie eine Funktion, welche $F\lambda$ auswertet, indem (bis auf $\tilde{S}_{\Pi\Pi}$) nur Teilgebietsmatrizen verwendet werden; siehe Anhang.

Geben Sie die benötigte Anzahl Iterationen und die Konditionszahlschätzung aus und plotten Sie die Teilgebietslösungen wie in der Programmieraufgabe auf dem 2. Übungsblatt.

Vergleichen Sie die Lösung mit der des globalen Systems $\hat{K}\hat{u} = \hat{b}$ (Norm der Differenz), wobei hier $\hat{K}$ die global assemblierte Steifigkeitsmatrix mit eliminierten Randwerten und $\hat{b}$ der zugehörige Lastvektor ist. Lösen Sie das globale System mit dem Backslash-Operator.

Lösen Sie nun zusätzlich das global assemblierte System mit Ihrer PCG-Implementierung und geben Sie auch hier die Norm der Differenz zur Lösung des direkten Lösers sowie die benötigte Anzahl Iterationen und die Konditionszahlschätzung aus.

Wählen Sie als Toleranz für das PCG-Verfahren 10^{-8} bezüglich des relativen Residuums und den Nullvektor als Startvektor.

b) Plotten Sie die Lösungen u_0, u_1 und u_2, welche zu den Iterierten λ_0, λ_1 und λ_2 aus dem PCG-Verfahren zur Lösung von $F\lambda = d$ gehören.

Allgemeine Hinweise zum Programmierteil

- Der Code **muss sinnvoll kommentiert** sein. Ein nicht kommentiertes Programm gilt als nicht erfolgreich bearbeitet.
- Das Programm muss ausführbar sein, ohne Änderungen am Code vornehmen zu müssen (d.h. ein Klick auf „Ausführen“ muss ausreichen). Schreiben Sie daher ein oder mehrere Skripte für die Teilaufgabe(n). Benennen Sie das Skript / die Skripte sinnvoll (z.B. `aufg1c.m`).
- Schreiben Sie bitte Funktionen in eigene Dateien und nicht in Skriptdateien (*Ausnahme:* anonyme Funktionen der Art `f = @(x) x.^2;`).
- Enthält Ihr Code mehrere Funktionen, so ist jede Funktion in eine eigene Datei zu schreiben. *Ausnahme:* Die Funktion wird ausschließlich von anderen Funktionen derselben Datei aufgerufen. In diesem Fall steht an oberster Stelle der Funktionsdatei die Funktion, welche von außerhalb (z.B. von einem Skript) aufgerufen wird.

Abgabe des Programmierteils

- Packen Sie Ihre Dateien in ein Archiv (Formate: .zip, oder .tar.gz) mit einem Dateinamen der Art:

`ueb01_nachname_vorname.zip`

- Den Quellcode schicken Sie bitte an die E-Mail-Adresse Ihrer Übungsgruppenleiter / Übungsgruppenleiterinnen, mit einem Betreff der Art:

  `Betreff: Uebung1, Nachname, Vorname`

- Geben Sie bitte immer eine **ausgedruckte Version** Ihrer Programmcodes mit den schriftlichen Aufgaben ab (→ Kasten), sofern dies in der Aufgabenstellung nicht eindeutig anders vermerkt wurde.

- Sofern es zur sinnvollen Lösung der Aufgabenstellung nötig ist, drucken Sie bitte auch die Ausgabe von Matlab aus. Dies sollte nicht zwei DIN-A4-Seiten überschreiten. Gleiches gilt für Grafiken.

Abgabe im entsprechenden Kasten in Raum 3.01 des Mathematischen Instituts.

- Theorie: Bis Mittwoch, 28. November 2018, 16:00 Uhr.

- Programmieraufgabe: Bis Mittwoch, 05. Dezember 2018, 16:00 Uhr.

Anhang - Implementierungshinweise

Listen und Indexabbildungen: Stellen sie analog zu `gamma` der vorherigen Programme nun logische Vektoren auf, welche die dualen bzw. primalen Knoten markieren (`dual`, `primal`). Zugehörig stellen Sie die Indexabbildungen *global* $\rightarrow$ *dual* bzw. *global* $\rightarrow$ *primal* auf; `mapDual`, `mapPi`. Anschließend stellen Sie (wie vorher `cGamma`) Cell-Arrays mit lokalen logischen Vektoren für die dualen und primalen Knoten auf; `cDual`, `cPrimal`. Zusätzlich stellen Sie noch lokale Listen für $B = (I, \Delta)$ auf; `cIDual`. Erstellen Sie nun noch Listen (Cell-Arrays), passend zu `cDual`, welche die zugehörigen Nummern der Lagrangeschen Multiplikatoren angeben (vgl. mit `cGammaMap`).

Sprungoperatoren: Die Sprungoperatoren `cB` lassen sich ähnlich einfach wie für das Streifengebiet aufstellen, da wir in den primalen Knoten keine Lagrangeschen Multiplikatoren benötigen. Man kann z.B. einen globalen Vektor definieren, diesen mit 1 initialisieren und anschließend über alle Gebiete iterieren. Man greift dann in jeder Iteration auf den globalen Vektor zu und schreibt die entsprechenden Werte für die Lagrangeschen Multiplikatoren des Teilgebiets in den lokalen Sprungoperator. Anschließend negiert man die verwendeten Werte. Mit den Listen `cDualMap{i}`, `find(cDual{i})` und dem Befehl

```
cB{i}=sparse(Zeilenindizes,Spaltenindizes,Werte,Zeilenanzahl,Spaltenanzahl);
```

kann man in wenigen Zeilen den Sprungoperator aufstellen.

Assemblierung: Assemblieren Sie wie bisher die lokalen Steifigkeitsmatrizen und Lastvektoren und eleminieren Sie die Randwerte. Stellen Sie nun $\tilde{K}_{\Pi\Pi}$ und $\tilde{f}_{\Pi}$ auf. Bedenken Sie, dass dies genauso funktioniert wie bei der Assemblierung der Steifigkeitsmatrix durch „Addieren“ der Elementsteifigkeitsmatrizen. Sie können hier mit `mapPi(l2g__sd{i}(cPrimal{i}))` die Indizes für das i-te Teilgebiet bezüglich der globalen Nummerierung von Π bestimmen. Addieren Sie die entsprechenden Einträge (`cPrimal{i}`) aus den lokalen Steifigkeitsmatrizen auf $\tilde{K}_{\Pi\Pi}$. Analog für $\tilde{f}_{\Pi}$.

Matrizen extrahieren: Bei FETI-DP betrachten wir die Knotenpartitionierung $B = (I, \Delta)$ und Π. Dementsprechend extrahieren Sie die folgenden Matrizen:

- $B^{(i)} \rightarrow B_B^{(i)} =$ `cB_B{i}`
- $K^{(i)} \rightarrow K_{BB}^{(i)} =$ `cK_BB{i}`
- $f^{(i)} \rightarrow f_B^{(i)} =$ `cb_B{i}`
- $K^{(i)} \rightarrow K_{\Pi B}^{(i)} =$ `cK_PiB{i}`

Achtung: Sie werden im Folgenden $K_{\Pi B}^{(i)}$ mehrfach benötigen und, nach Notation der Vorlesung, bezieht sich Π hier auf alle primalen Knoten und nicht nur auf die des Teilgebiets. In `cPrimal{i}` stehen natürlich nur die „lokalen“ primalen Knoten; daher wird $K_{\Pi B}^{(i)}$ bei Ihnen die falsche Dimension haben. Umgehen Sie dies, indem Sie zunächst mit

```
cK_PiB{i}=sparse(AnzahlPrimalKnotenGlobal,AnzahlInnereUndDualeKnotenLokal);
```

initialisieren und, wie beim Assemblieren von $\tilde{K}_{\Pi\Pi}$, an die ensprechenden Stellen die Werte aus $K^{(i)}$ schreiben.

Berechnung des Schurkomplements: Mit der letzten Bemerkung sollte dies nun kein Problem darstellen. Stellen Sie nicht K_{BB} auf, sondern schreiben Sie $\tilde{K}_{\Pi B} K_{BB}^{-1} \tilde{K}_{B\Pi}$ wie in der Vorlesung als Summe von Produkten lokaler Matrizen.

Funktionen definieren: Definieren Sie drei Funktionen für die Operationen

- $B_B K_{BB}^{-1} \tilde{K}_{B\Pi} \cdot x$,
- $\tilde{K}_{\Pi B} K_{BB}^{-1} B_B^T \cdot x$,
- $F \cdot x$.

Wie bisher auch: Schreiben Sie die Operationen als Summe lokaler Matrix-Vektor-Operationen.

Berechnung der rechten Seite d und Lösung mit PCG: Mit den oben definierten Funktionen können Sie nun die rechte Seite d aus $F\lambda = d$ explizit aufstellen. Erzeugen Sie ein Handle auf die Funktion F, z.B. mit

```
hF = @(lambda) F(cB_B,cK_BB,cK_PiB,S_PiPi,lambda);
```

und übergeben Sie dies Ihrer PCG-Implementierung.

Rückwärts-Substitution: Schreiben Sie nun noch eine Funktion, welche aus λ die lokalen Lösungen $u^{(i)}$ berechnet. Alle notwendigen Matrizen und Funktionen sollten dafür vorhanden sein.

Aufgabe b) Verwenden Sie die Funktion zur Rückwärtssubstitution, indem Sie Ihrer PCG-Implementierung ein Function-Handle auf eine Funktion `plotiter` übergeben, welche die Rückwärtssubstitution aufruft und die Teilgebietslösungen plottet.

```
ploth = @(lambda,iter) plotiter(lambda,iter,cB_B,cK_BB,cK_PiB,cb_B,mapPi, ...
                                l2g__sd,cPrimal,cIDual,S_PiPi,tri__sd,x__sd);
```

Im PCG-Verfahren müssen Sie dann nur noch `ploth(lambda,iter)` aufrufen.